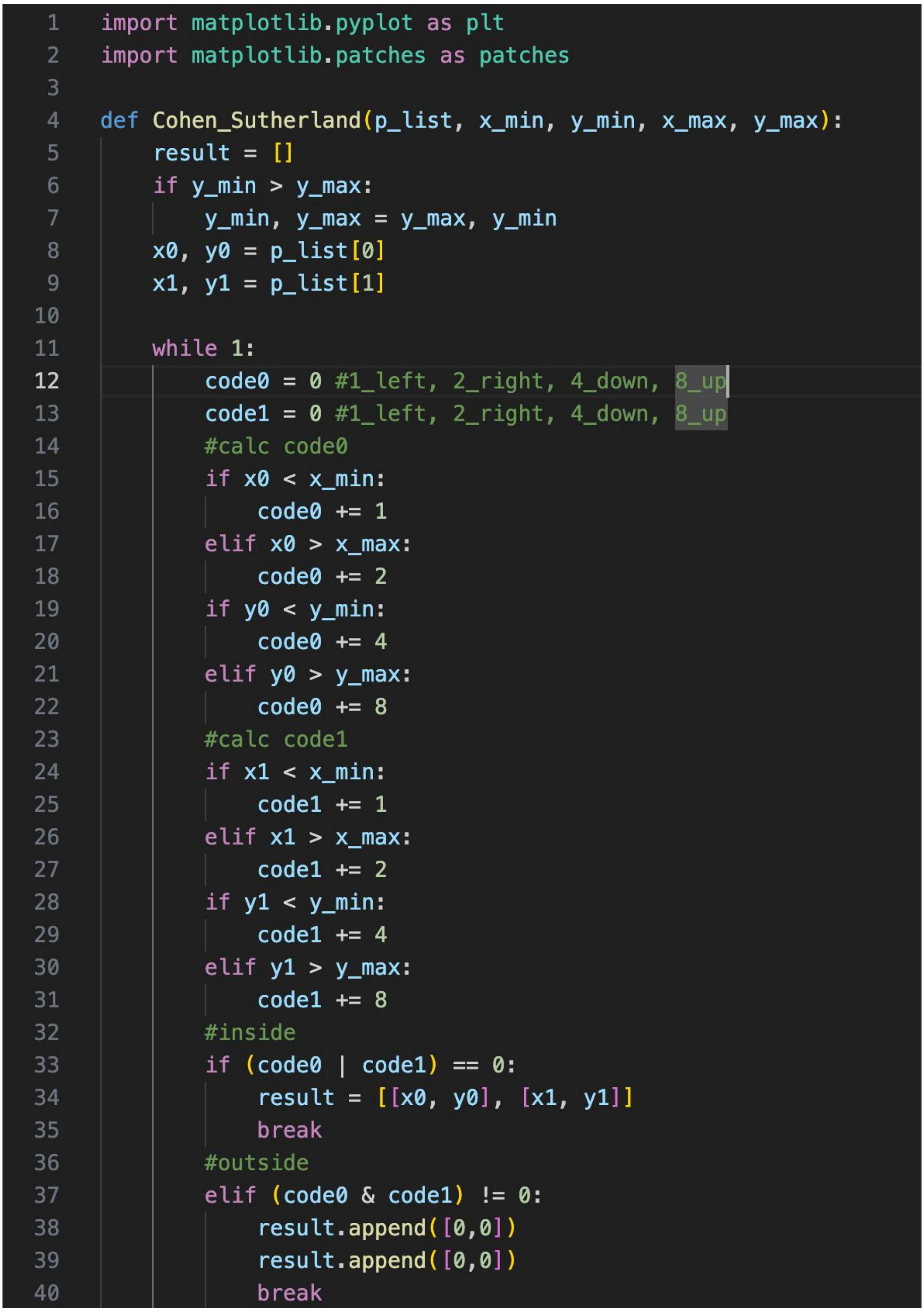
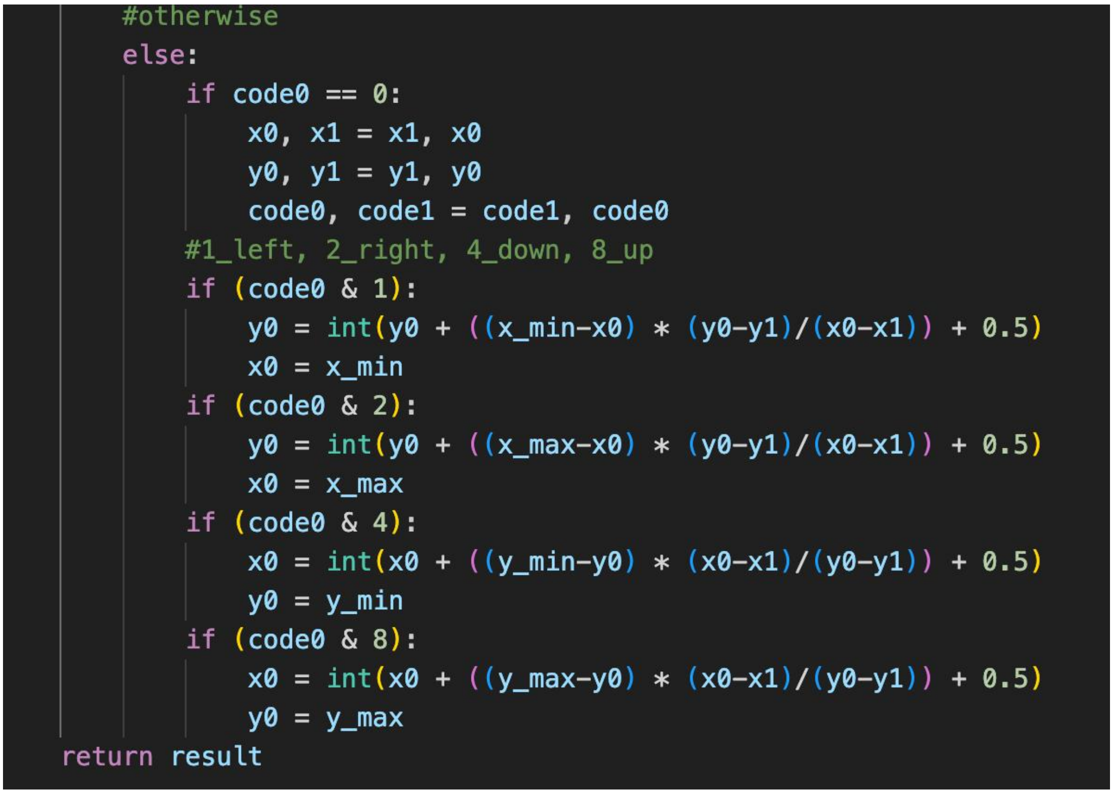
Assignment5

22124187 Zhang Linyihan

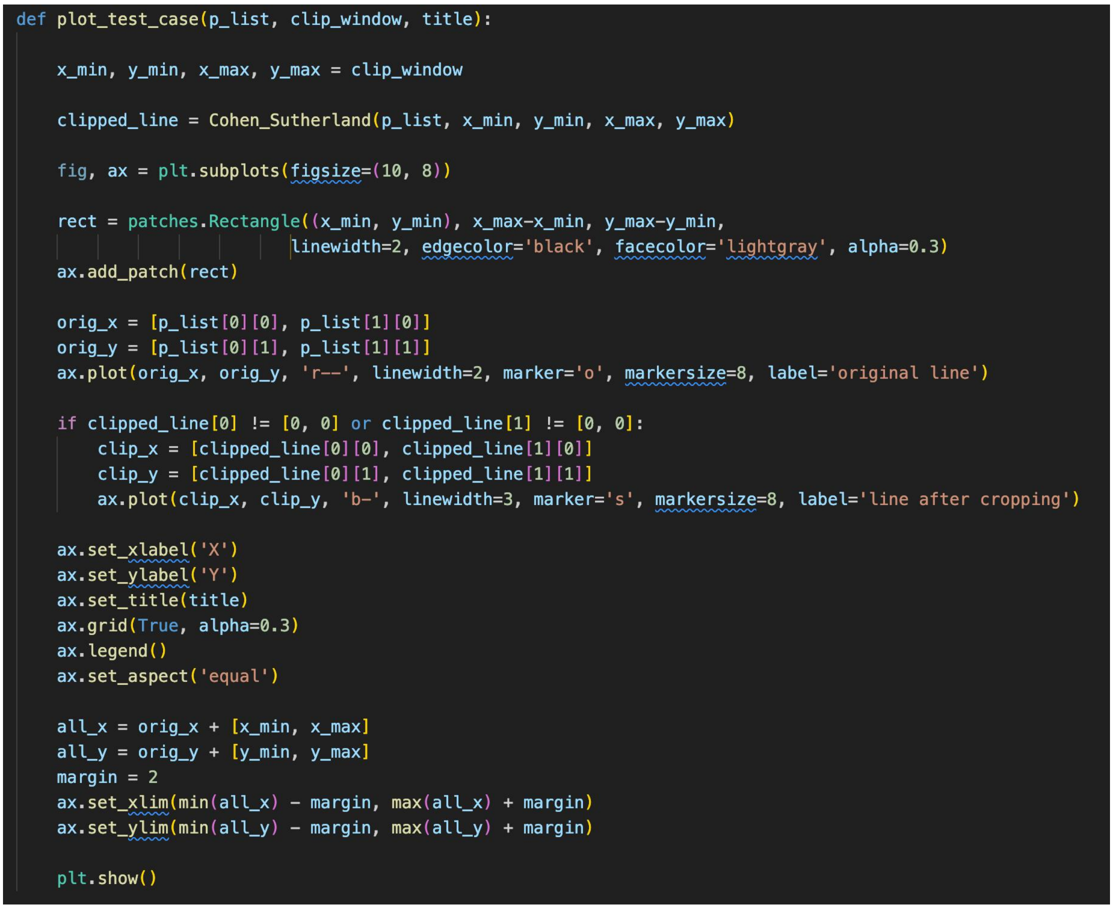
Cohen-Sutherland line clipping algorithm



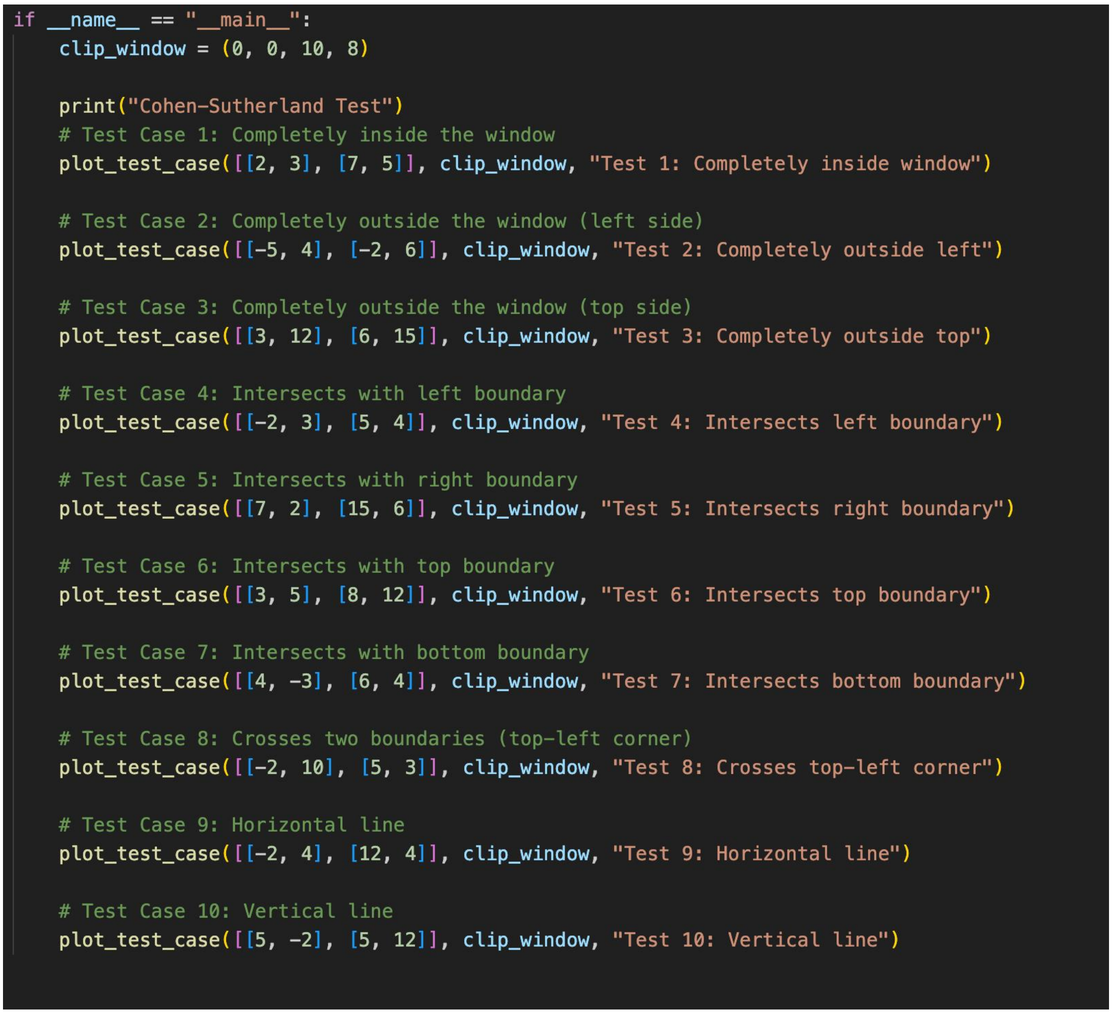
```python
import matplotlib.pyplot as plt
import matplotlib.patches as patches

def Cohen_Sutherland(p_list, x_min, y_min, x_max, y_max):
    result = []
    if y_min > y_max:
        y_min, y_max = y_max, y_min
    x0, y0 = p_list[0]
    x1, y1 = p_list[1]

    while 1:
        code0 = 0 #1_left, 2_right, 4_down, 8_up
        code1 = 0 #1_left, 2_right, 4_down, 8_up
        #calc code0
        if x0 < x_min:
            code0 += 1
        elif x0 > x_max:
            code0 += 2
        if y0 < y_min:
            code0 += 4
        elif y0 > y_max:
            code0 += 8
        #calc code1
        if x1 < x_min:
            code1 += 1
        elif x1 > x_max:
            code1 += 2
        if y1 < y_min:
            code1 += 4
        elif y1 > y_max:
            code1 += 8
        #inside
        if (code0 | code1) == 0:
            result = [[x0, y0], [x1, y1]]
            break
        #outside
        elif (code0 & code1) != 0:
            result.append([0,0])
            result.append([0,0])
            break
```



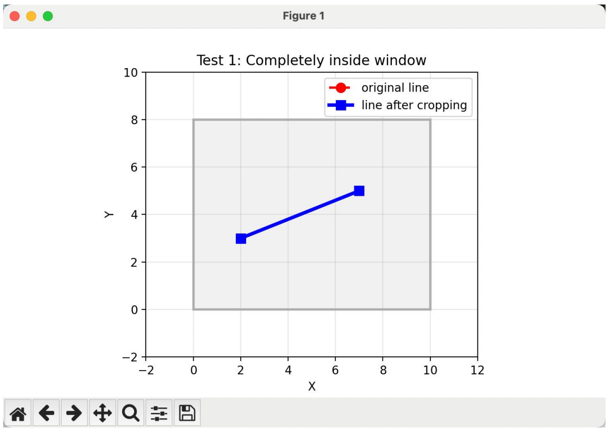
```python
        #otherwise
        else:
            if code0 == 0:
                x0, x1 = x1, x0
                y0, y1 = y1, y0
                code0, code1 = code1, code0
            #1_left, 2_right, 4_down, 8_up
            if (code0 & 1):
                y0 = int(y0 + ((x_min-x0) * (y0-y1)/(x0-x1)) + 0.5)
                x0 = x_min
            if (code0 & 2):
                y0 = int(y0 + ((x_max-x0) * (y0-y1)/(x0-x1)) + 0.5)
                x0 = x_max
            if (code0 & 4):
                x0 = int(x0 + ((y_min-y0) * (x0-x1)/(y0-y1)) + 0.5)
                y0 = y_min
            if (code0 & 8):
                x0 = int(x0 + ((y_max-y0) * (x0-x1)/(y0-y1)) + 0.5)
                y0 = y_max
    return result
```

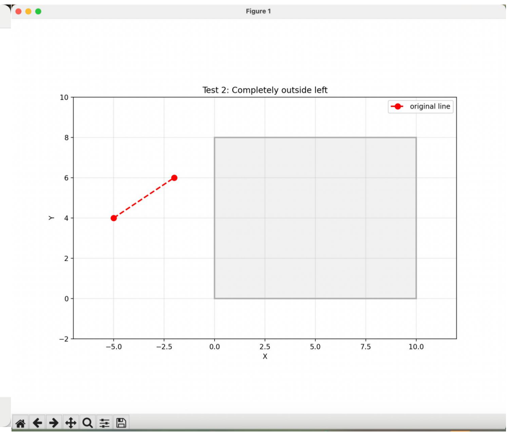


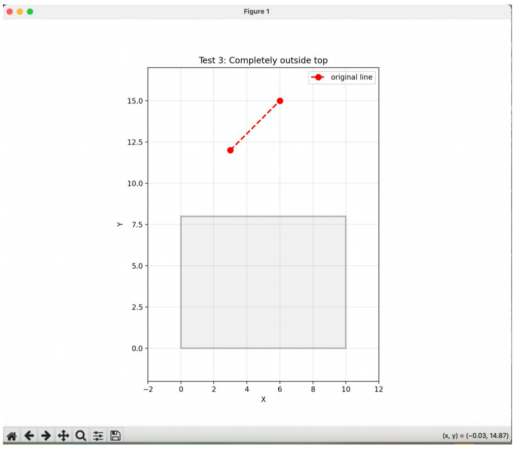
```python
def plot_test_case(p_list, clip_window, title):

    x_min, y_min, x_max, y_max = clip_window

    clipped_line = Cohen_Sutherland(p_list, x_min, y_min, x_max, y_max)

    fig, ax = plt.subplots(figsize=(10, 8))

    rect = patches.Rectangle((x_min, y_min), x_max-x_min, y_max-y_min,
                            linewidth=2, edgecolor='black', facecolor='lightgray', alpha=0.3)
    ax.add_patch(rect)

    orig_x = [p_list[0][0], p_list[1][0]]
    orig_y = [p_list[0][1], p_list[1][1]]
    ax.plot(orig_x, orig_y, 'r--', linewidth=2, marker='o', markersize=8, label='original line')

    if clipped_line[0] != [0, 0] or clipped_line[1] != [0, 0]:
        clip_x = [clipped_line[0][0], clipped_line[1][0]]
        clip_y = [clipped_line[0][1], clipped_line[1][1]]
        ax.plot(clip_x, clip_y, 'b-', linewidth=3, marker='s', markersize=8, label='line after cropping')

    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_title(title)
    ax.grid(True, alpha=0.3)
    ax.legend()
    ax.set_aspect('equal')

    all_x = orig_x + [x_min, x_max]
    all_y = orig_y + [y_min, y_max]
    margin = 2
    ax.set_xlim(min(all_x) - margin, max(all_x) + margin)
    ax.set_ylim(min(all_y) - margin, max(all_y) + margin)

    plt.show()
```



```python
if __name__ == "__main__":
    clip_window = (0, 0, 10, 8)

    print("Cohen-Sutherland Test")
    # Test Case 1: Completely inside the window
    plot_test_case([[2, 3], [7, 5]], clip_window, "Test 1: Completely inside window")

    # Test Case 2: Completely outside the window (left side)
    plot_test_case([[-5, 4], [-2, 6]], clip_window, "Test 2: Completely outside left")

    # Test Case 3: Completely outside the window (top side)
    plot_test_case([[3, 12], [6, 15]], clip_window, "Test 3: Completely outside top")

    # Test Case 4: Intersects with left boundary
    plot_test_case([[-2, 3], [5, 4]], clip_window, "Test 4: Intersects left boundary")

    # Test Case 5: Intersects with right boundary
    plot_test_case([[7, 2], [15, 6]], clip_window, "Test 5: Intersects right boundary")

    # Test Case 6: Intersects with top boundary
    plot_test_case([[3, 5], [8, 12]], clip_window, "Test 6: Intersects top boundary")

    # Test Case 7: Intersects with bottom boundary
    plot_test_case([[4, -3], [6, 4]], clip_window, "Test 7: Intersects bottom boundary")

    # Test Case 8: Crosses two boundaries (top-left corner)
    plot_test_case([[-2, 10], [5, 3]], clip_window, "Test 8: Crosses top-left corner")

    # Test Case 9: Horizontal line
    plot_test_case([[-2, 4], [12, 4]], clip_window, "Test 9: Horizontal line")

    # Test Case 10: Vertical line
    plot_test_case([[5, -2], [5, 12]], clip_window, "Test 10: Vertical line")
```

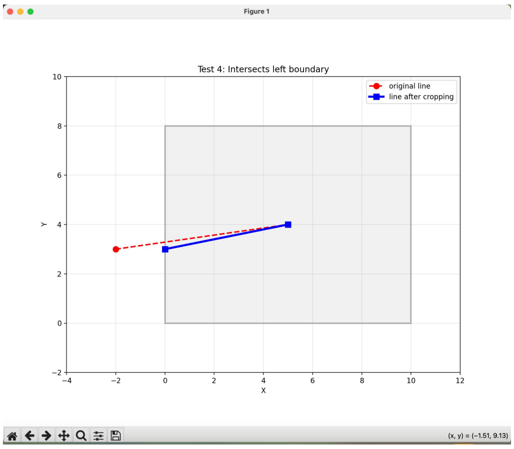






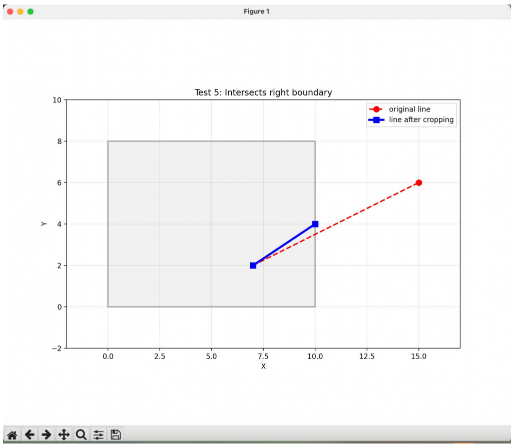
Test 3: Completely outside top

(x, y) = (−0.03, 14.87)

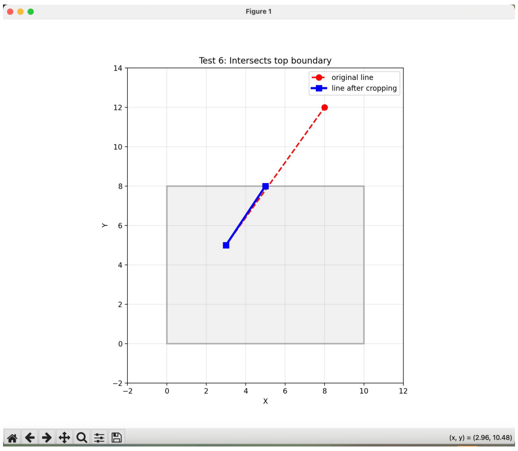


Test 4: Intersects left boundary

(x, y) = (−1.51, 9.13)

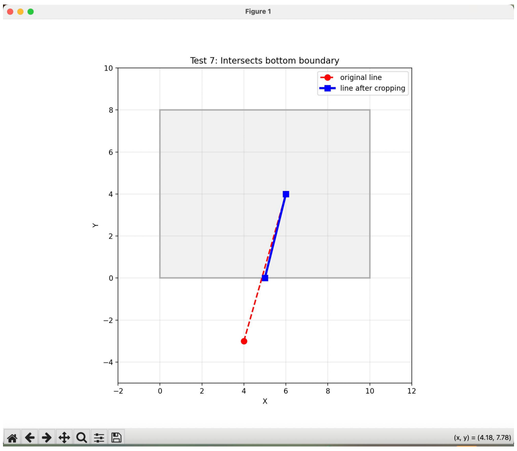


Test 5: Intersects right boundary



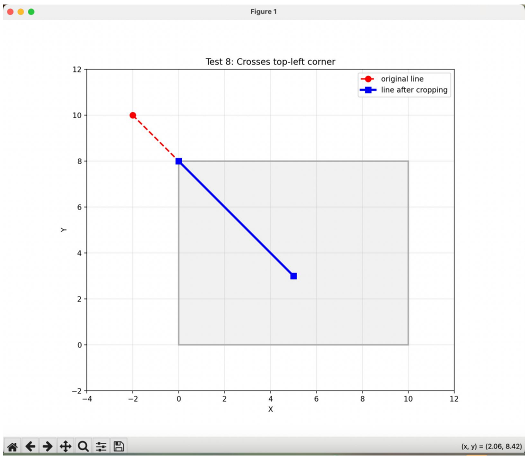
Test 6: Intersects top boundary

(x, y) = (2.96, 10.48)



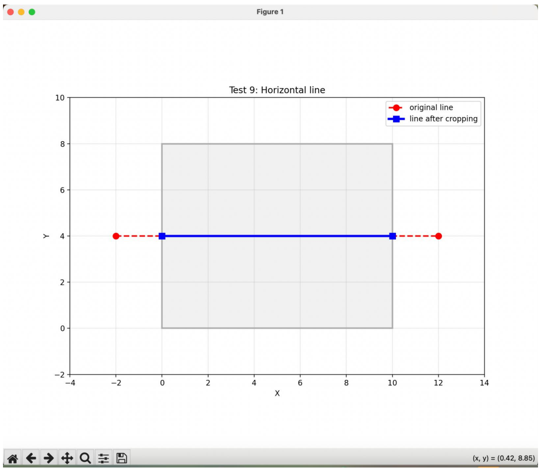
Test 7: Intersects bottom boundary

(x, y) = (4.18, 7.78)



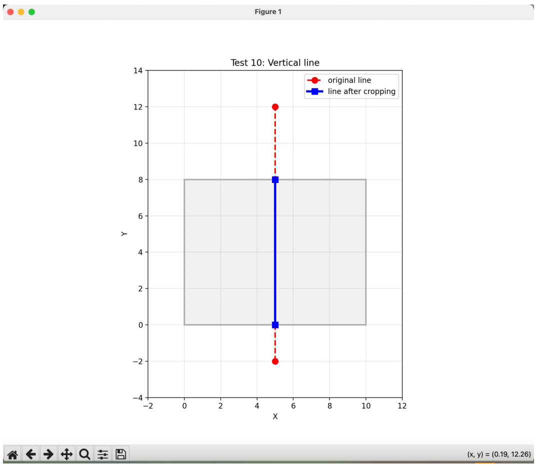
Test 8: Crosses top-left corner

(x, y) = (2.06, 8.42)



Test 9: Horizontal line

(x, y) = (0.42, 8.85)



Test 10: Vertical line

(x, y) = (0.19, 12.26)